UNIT 14 SYSTEM ANALYSIS AND DESIGN

Structure

14.1 INTRODUCTION

From the inception of an idea for a software system, until it is implemented and delivered to a customer, and even after that, the system undergoes gradual development and evolution. The software is said to have a life cycle composed of several phases. Each of these phases results in the development of either a part of the system or something associated with the system, such as a test plan or a user manual. In the traditional life cycle model, each phase has well-defined starting and ending points, with clearly identifiable deliverables to the next phase.

14.2 OBJECTIVES

After reading this unit, you should be able to:

- Describe traditional systems life cycles;
- Identify different phases of system life cycle;
- Explain the process of system analysis;
- Describe the conceptual basis of system design; and
- Know about Implementation and maintenance of software.

14.3 TRADITIONAL SYSTEM LIFE CYCLE

The goal of the traditional system life cycle is to keep the project under control and assure that the information system produced, satisfies the requirements. The traditional system life cycle divides the project into a series of steps, each of which has distinct deliverables, such as documents or computer programs. This is known as the systems development life cycle (SDLC). The deliverables are related because each subsequent step builds on the conclusions of previous steps. This has been shown in Figure 14.1. Some deliverables are oriented toward the technical staff, whereas others are directed toward or produced by users and mangers. The latter ensure that users and their management are included in the system development process.

Fig. 14.1: Phases of SDLC

Source: *Alter Steven, (1999) Information Systems (A management perspective), Pearson Education*

Although there is general agreement about what needs to be done in the traditional system life cycle, different authors name individual steps and deliverables differently. Many versions of the traditional system life cycle emphasize the building or software and de-emphasize what happens in the organization before and after software development. Because this unit is directed at business professionals, its version of the traditional system life cycle emphasizes implementation and operation in the organization in addition to software development.

14.3.1 Initiation

The initiation phase may begin in many different ways. A user may work with the IS staff to produce a written request to study a particular business problem. The IS staff may discover an opportunity to use information systems beneficially and then try to interest users. A top manager may notice a business problem and ask the head of IS to look into it. A computer crash or other operational problem may reveal a major problem that can be patched temporarily but requires a larger project to fix it completely. Regardless of how this phase begins, its goal is to analyze the scope and feasibility of a proposed system and to develop a project plan. This involves two steps, the feasibility study and project planning, which produce the functional specification and a project plan.

The feasibility study is a user-oriented overview of the proposed information system's purpose and feasibility. A system's feasibility is typically considered from economic, technical, and organizational viewpoints.

- Economic feasibility involves question such as whether the firm can afford to build the information system, whether its benefits should substantially exceed its costs, and whether the project has higher priority than other projects that might use the same resources.

- Technical feasibility involves question such as whether the technology needed for the information system exists and whether the firm has enough experience using that technology.

- Organizational feasibility involves questions such as whether the information system has enough support to be implemented successfully, whether it brings as excessive amount of change, and whether the organization is changing too rapidly to absorb it.

If the information system appears to be feasible, the initiation phase produces a functional specification and a project plan. The functional specification explains the important of the business problem; summarizes changes in business processes; and estimates the project's benefits, costs, and risks. The project plan breaks the project into sub-projects with start and completion times. It also identifies staffing, resource requirements, and dependencies between project steps.

The functional specification is approved by both user and IS personnel. It clarifies the purpose and scope of the proposed project by describing the business processes that will be affected and how they will be performed using the system. Functional specifications once consisted primarily of prose. With the advent of diagramming tools such as data flow have become much easier to read and understand. These visual representations help parts of the system will play. Functional specifications typically do not explain exactly what data, reports, or data entry screens will be included. This more detailed description is produced in the development phase.

Fig. 14.2: Steps in Development Phase of System Life Cycle

Source: *Alter Steven, (1999) Information Systems (A management perspective), Pearson Education*

14.3.2 Development

The development phase creates computer programs (with accompanying user and programmer documentation) plus installed hardware that accomplishes the data processing described in the functional specification. This is done through a process of successive refinement in which the functional requirements are translated into computer programs and hardware requirements. The purpose of the various steps and deliverables in the development phase is to ensure that the system accomplishes the goals explained in the functional specification. These steps are summarized in *Figure 14.2*.

The first step in the development phase is the detailed requirements analysis, which produces a user-oriented description of exactly what the information system will do. This step is usually performed by a team including user representative and the IS department.

It produces a document called the external specification. Building on the functional specification, the external specification shows the data input screens and major reports and explains the calculations that will be automated. It shows what information system users will see, rather than explaining exactly how the computer will perform the required processing. Users reviewing this document focus on whether they understand the data input screens, reports, and calculations, and whether these will support the desired business process. By approving the external specification, the users and IS staff signify their belief that the information system will accomplish what they want.

The next step is internal system design, in which the technical staff decides how the data processing will be configured on the computer. This step produces the internal specification, a technical blueprint for the information system. It documents the computer environment the system will operate in, the detailed structure and content of the database, and the inputs and outputs of all programs and subsystems. Users do not sign off on the internal specification because it addresses technical system design issues. Instead, the IS staff signs off that the internal specification accomplishes the functions called for in the external specification the users have approved.

Thus far the discussion has focused on software. Because the software will work only if there is hardware for it to run on, and essential step in the development phase is hardware acquisition and installation. For some information systems, this is not an issue because it is a foregone conclusion that existing hardware will be used. Other systems require a careful analysis to decide which hardware to acquire, how to acquire it most economically, where to put it, and how to install it by the time it is indeed. Factors considered in hardware acquisition decisions include compatibility with existing hardware and software, price, customer service, and compatibility with long-term company plans. Computer hardware can be purchased or rented through a variety of financing arrangements, each with its own tax consequences. A firm’s finance department usually makes the financing arrangements for significant hardware purchases. Especially if new computer hardware requires a new computer room, lead times for building the room, installing the electricity and air conditioning, and installing the computer may be important factors in the project plan.

In firms with large IS staffs; users rarely get involved with the acquisition, installation, and operation of computer hardware. Much as with telephone systems, users expect the hardware to be available when needed and complain furiously whenever it goes down. This is one reason computer hardware managers sometimes consider their jobs thankless.

Programming is the creation of the computer code that performs the calculations, collects the data, and generates the reports. It can usually proceed while the hardware is being acquired and installed. Programming includes the coding, testing, and documentation of each program identified in the internal specification. Coding is what most people think of as programming. The testing done during the programming step is often called unit testing, because it treats the programs in isolation.
The documentation of each program starts with the explanation from the internal specification and includes comments about technical assumptions made in writing the program, plus any subtle, non-obvious processing done by the program.

A number of improvements in programming methods have made programming faster and more reliable. Structured programming is often used to make the programs more consistent, easier to understand and less error prone. Fourth generation languages (4GLs) also expedite programming for some systems. However, as should be clear from all of the steps leading up to coding and following coding, coding often accounts for less than 20% of the work in developing a system. This is one of the reasons 4GLs and other improved programming tools do not drastically shrink the system life cycle for large systems, even when they slash programming time.

Documentation is another activity that can proceed in parallel with programming and hardware acquisition. Both user and technical documentation is completed from the material that already exists. The functional specification and external specification are the basis for the user documentation, and the internal specification and program documentation are the basis for the programmer documentation. With the adoption of Computer Aided Software Engineering (CASE) tools, more of the documentation is basically a compilation of data and diagrams already stored on a computer. Additional user documentation is usually required, however, because different users need to know different things depending on their roles. People who perform data entry tasks need to understand the data entry procedures and what the data mean; people who use data from the system need to understand what the data mean and how to retrieve and analyze data, but do not need to know much about data entry details.

After the individual programs have been tested, the entire information system must be tested to ensure that the programs operate together to accomplish the desired functions. This is called the system testing, or integration testing. System tests frequently uncover inconsistencies among programs as well as inconsistencies in the original internal specification. These must be reconciled and the programs changed and retested. One of the reasons for Microsoft’s “synch and stabilize” method is to eliminate the surprises and extensive network that might occur if system testing showed that programs were incompatible. Although system testing may seem an obvious requirement, inadequate system testing has led to serious problems.
For example, a new trust accounting system put into operation prematurely by Bank of America on March 1, 1987, lost data and fell months behind in generating statements for customers. By January 1988, 100 institutional customers with $ 4 billion in assets moved to other banks, several top executives resigned, and 2.5 million lines of code were scrapped.

An important part of testing is the creations of a testing plan, a precise statement of exactly how the information system will be tested. This plan includes the data that will be used for testing. Creating a testing plan serves many purposes. It encourages careful thought about how the system will be tested. In addition, a thorough plan increases the likelihood that all foreseeable contingences will be considered and that the testing will catch more of the bugs in the system.

It should be clear that the development phase for a large information system is a complex undertaking, quite different from sitting down at a personal computer and developing a small spreadsheet model. Explicitly separating out all the steps in the development phase helps to ensure that the information system accomplishes the desired functions and is debugged. Such an elaborate approach is needed because the system is a tool of an organization rather than an individual. An individual producing a spreadsheet is often typing to solve a current problem with no intention to use the spreadsheet next month, much less that someone else will need to decipher and modify it next year. In contrast, the traditional system life cycle assumes that the information system may survive for years, may be used by people who were not involved in its development, and may be changed repeatedly during that time by people other than the original developers. The steps in the traditional life cycle try to make the long-term existence of the information system as efficient error-free as possible.

14.3.3 Implementation

Implementation is the process of putting a system into operation in an organization. *Figure 14.3* shows that it starts with the end product of the development phase, namely, a set of computer programs that run correctly on the computer, plus accompanying documentation. This phase begins with implementation planning, the process of creating plans for training, conversion, and acceptance testing.

Fig. 14.3: Steps in the Implementation of System Life Cycle

Source: *Alter Steven, (1999) Information Systems (A management perspective), Pearson Education*

The training plan explains how and when the user will be trained. The conversion plan explains how and when the organization will convert to new business processes. The acceptance-testing plan describes the process and criteria for verifying that the information system works properly in supporting the improved work system.

Training is the process of ensuring that system participants know what they need to know about both the work system and the information system. The training format depends on user backgrounds and the purpose and features of both the work system and the information system. Users with no computer experience may require special training. Training for frequently used transaction processing systems differs from training for data analysis systems that are used occasionally. Information systems performing diverse functions require more extensive training than systems used repetitively for new functions. Training manuals and presentations help in the implementation system. After the previous methods have receded into history, other types of training material are more appropriate.

Following the training comes the carefully planned process of conversion from the old business processes to new ones using the new information system. Conversion is often called cutover or changeover. It can be accomplished in several ways, depending on the nature of the work and the characteristics of the old and new systems. One possibility is to simply choose a date, shut off the old information system, and turn on the new one while hoping that the work system will operate as intended. This is risky, though, because it does not verify that the information system will operate properly and that the users understand how to use it.

Consider the following example: The State of California installed an optical disk system to streamline the process of doing title searches (establishing ownership and identifying indebtedness on a property) for borrowers who wished to purchase property. Previously, there was a 2 to 3 week delay between the borrower’s loan request and the bank’s receipt of a confirmation that the title was clear. The new system was to reduce this delay to 2 days. Both the vendor and several state officials recommended that the existing manual system remain in full operation during the conversion in case of problems. However, the Secretary of Finance rejected the request for an additional $2.4 million, and the manual system was simply shut down when the optical disk system came up. Unfortunately, software bugs plagued the new system, and the resulting logjam of 50,000 loan requests delayed title searches for up to 10 weeks. The new system was shut down for repair, and the old manual system reinstated. The Assistant Secretary of State said that some banks almost went out of business because of the slow turnaround.

To minimize risk and wasted effort, most conversions occur in stages, which can be done in several ways. A phased approach uses the new information system and work system for a limited subset of the processing while continuing to use old methods for the rest of the processing. If something goes wrong, the part of the business using the new system can switch back to the old system. The simultaneous use of the old system and the new system is called running in parallel. Although this involves double record keeping for a while, it verifies that the new information system operates properly and helps the users understand how to use it effectively within the new work system.

Conversions from one computerized system to another are often far more difficult than users anticipate. Part of the problem is that computerized data in the old system must be converted into the formats used by the new system. In consistencies between the two systems frequently lead to confusion about whether the data in

either system are correct. Furthermore, programs that convert the data from one system to another may have their own bugs, thereby adding to confusion and delays.

Conversion requires careful planning because people who don't want the new system and use the problems as an opportunity to complain can blow even minor problems out of proportion. For these reasons, it is often wise to do a pilot implementation with a small group of users who are enthusiastic about the system improvements. Ideally, their enthusiasm will motivate them to make the effort to learn about the changes and to forgive minor problems. After a pilot implementation demonstrates that the new information system works, it is usually much easier to motivate everyone else (including the skeptics) to start using it.

Acceptance testing is testing of the information system by the users as it goes into operation. Acceptance testing is important because the information system may not fit, regardless of what was approved and signed off in the external specification. The business situation may have changed; the external specification may reflect misunderstandings; the development process may have introduced errors; or the implementation may have revealed unforeseen problems. For all these reasons, it makes sense to include an explicit step of deciding whether the information system is accepted for ongoing use. If it doesn't fit user needs, for whatever reason, installing it without changes may lead to major problems and may harm the organization instead of helping. Acceptance testing also solidifies user commitment because it gets people in the user organization to state publicly that the system works.

The post-implementation audit is the last step in the implementation phase, even though it occurs after the new system has been in operation for a number of months. Its purpose is to determine whether the project has met its objectives for costs and benefits and to make recommendations for the future. This is also an opportunity to identify what the organization can learn from the way the project was carried out.

14.3.4 Operations and Maintenance

The operation and maintenance phase starts after the users have accepted the new system. This phase can be divided into two activities: (1) ongoing operation and support, and (2) maintenance. Unlike the other steps in the life cycle, these steps continue throughout the system's useful life. The end of a system's life cycle is its absorption into another system or its termination.

Ongoing operation and support is the process of ensuring that the technical system components continue to operate correctly and that the users use it effectively. This process is similar to the process of making sure a car or building operates well. It works best when a person or group has direct responsibility for keeping the information system operating. This responsibility is often split, with the technical staff taking care of computer operations and a member of the user organization ensuring that users understand the system and use it effectively.

Day-to-day computer operations typically include scheduled events such as generating summary reports from management and backups of the database. The operations manual specifies when these jobs should be done. For transaction processing systems essential to the operation of the business, a member of the technical staff also monitors computer-generated statistics related to response times, program run times, disk space utilization, and similar factors to ensure the programs are running efficiently.

When the database becomes too full, or when response times start to increase, the technical configuration of the information system must be changed. This is done by allocating more disk space, unloading (backing up onto tape or discarding) data that are not current, or changing job schedules.

Maintenance is the process of modifying the information system over time. As users gain experience with a system, they discover its shortcomings and usually suggest improvements. The shortcomings may involve problems unrelated to the information system or may involve ways that the information system might do more to support the work system, regardless of the original intentions. Some shortcomings are bugs. Important shortcomings must be corrected if users are to continue using an information system enthusiastically.

Handling enhancement requests and bug fix requests is both a technical challenge and a delicate political issue for IS departments. The technical challenge is ensuring that changes don't affect other parts of the system in unanticipated ways. The traditional life cycle helps here because documentation and internal design methods enforce modularization and make it easier to understand the scope and impact of changes.

The political issue for most IS departments are their inability to support even half of the enhancement requests they receive. For new or inadequately planned information systems, some departments have more enhancement requests than they can even analyze. In this environment, it requires both technical and political skill to keep users satisfied. Users are often frustrated by how long it takes to make changes.

What might seem to be a simple change to a person who "programs" spreadsheet is often vastly more complex in a large information spawn changes in several levels of documentation.

The steps in each of the four phases of the traditional system life cycle have now been introduced. Table 14.1 outlines the steps in each phase and makes two major points in addition to the details it presents. First it shows that users are highly involved in three of the four phases. In other words, building information systems is not just technical work done by the technical staff. It also shows that each step has specific deliverables that document progress to date and help keep the project under control.

The traditional system life cycle is a tightly controlled approach designed to reduce the likelihood of mistakes or omissions? Despite its compelling logic, it has both advantages and disadvantages. Adherence to fixed deliverables and signoffs improves control but guarantees a lengthy process. Having specific deliverables due at specific times makes it schedule of deliverables sometimes takes on a life of its own and seems as important as the real project goals.

When merely going through the motions of producing deliverables on schedule, participants may be tempted to turn in work that is incomplete and to approve documents they do not truly understand.

The traditional system life cycle is the standard against which other approaches are compared. Project managers who want to bypass some of its steps still need a way to deal with the issues they raise.

Table 14.1: Step and Deliverables in the Traditional System Life Cycle

Phase/Step	Degree of user participation	Key deliverable, plan or document	Key participants
Initiation			
* Feasibility study	High	Functional specification	User representa tives, management, and technical staff
* Project planning	Medium	Project plan	User representatives, management, and technical staff
Development			
*Detailed requirements analysis	High	External specification	User representatives, management, and technical staff
* Internal system design	None	Internal specification	Programmers and technical staff
* Hardware acquisition and installation	None	Hardware plan Hardware operational	Technical staff
* Programming	None	Individual programs debugged	Programmers
* Documentation	Medium	User and programmer documentation	Technical staff and users
* System testing	Medium	Test plan Completed system test	Programmers and users
Implementation			
* Implementation planning	High	Implementation plan	Training, users, and management
* Training	High	Training materials	Trainers and users
* Conversion	High	System in use	Users and project team
* Accepting testing	High	System accepted	Users and project team
* Post implementation audit	High	Audit report	Users and management
Operation and Maintenance			
* Ongoing operation and support	Low	Operations manual	Technical staff
	Low	Usage statistics	Technical staff & user
	High	Enhancement requests and bug fix requests	Technical staff and users
* Maintenance	Medium	Maintenance plan	Technical staff and users
* Absorption or termination	-	-	-

Activity A

Take a business problem concerning your top manager and the IS department. Propose a system, which can be put on in such cases. Analyze the scope and feasibility of the proposed system and develop a project plan.

.....

.....

.....

.....

.....

14.4 SYSTEMS LIFE CYCLE

The systems life cycle is the oldest method for building information systems and is still used today for complex medium or large systems projects. This methodology assumes that an information system has a life cycle similar to that of any living organism, with a beginning, middle, and an end. The life cycle for information system has six stages: project definition, systems study, design, programming, installation, and post-implementation. Each stage consists of basic activities that must be performed before the next stage can begin.

The life cycle methodology is a very formal approach to building systems. It partitions the systems development process into distinct stages and develops an information system sequentially, stage by stage. The life cycle methodology also has a very formal division of labor between end users and information systems specialists. Technical specialists such as systems analysts and programmers are responsible for much of the systems analysis, design, and implementation work; end users are limited to providing information requirements and reviewing the work of the technical staff. Formal sign-offs or agreements between and users and technical specialists are required as each stage is completed.

Product or output of each stage of the life cycle that is the basis for such sign-offs. The project definition stage results in a proposal for the development of a new system. The systems study stage provides a detailed systems proposal report outlining alternative solutions and establishing the feasibility of proposed solutions. The design stage results in a report on the design specifications for the system solution that is selected. The programming stage results in actual software code for the system. The installation stage outputs the results of tests to assess the performance of the system. The post-implementation stage concludes with a post-implementation audit to measure the extent to which the new system has met its original objectives. We now describe the stages of the life cycle in detail.

14.4.1 Stages of the Systems Life Cycle

The project definition stage tries to answer the questions, “Why do we need a new system project?” and “What do we want to accomplish?” This stage determines whether the organization has a problem and whether that problem can be solved by building a new information system or by modifying an existing one. If a system project is called for, this stage identifies its general objectives, specifies the scope of the project, and develops a project plan that can be shown to management.

The systems study stage analyzes the problems of existing systems (manual or automated) in detail, identifies objectives to be attained by a salutation to these problems, and describes alternative solutions. The systems study stage examines the feasibility of each solution alternative for review by management. This stage tries to answer the questions, “What doe the existing systems do?” “What are their strengths, weakness, trouble spots, and problems?” “What user information requirements must be met by the solution?” “What alternative solution options are feasible?” “What are their costs and benefits?”

Answering these questions requires extensive information gathering and research; sifting through documents, reports, and work papers produced by existing systems; observing how these systems work; polling users with questionnaires; and conducting interviews. All of the information gathered during the systems study phase will be used to determine information system requirements. Finally, the systems study stage describes in detail the remaining life cycle activities and the tasks for each phase.

The design stage produces the logical and physical design specification for the solution. Because the life cycle emphasizes formal specifications and paperwork, many of the design and documentation tools, such as data flow diagrams, structure charts, or system flowcharts are likely to be utilized.

The programming stage translates the design specifications produced during the design stage into software program code. Systems analysts work with programmers to prepare specifications for each program in the system. These program specifications describe what each program will do, the type of programming language to be used, inputs and outputs, processing logic, processing schedules, and control statements such as those for sequencing input data. Programmers write customized program code typically using a conventional third-generation programming language such as COBOL or FORTRAN or a high-productivity fourth-generation language. Since large systems have many programs with hundreds of thousands of lines of program code, entire teams of programmers may be required.

The installation stage consists of the final steps to put the new or modified system into operation: testing, training, and conversion. The software is tested to make sure it performs properly from both a technical and a functional business standpoint. Business and technical specialists are trained to use the new system. A formal conversion plan provides a detailed schedule of all of the activities required to install the new system, and the old system is converted to the new one.

The post-implementation stage consists of using and evaluating the system after it is installed and is in production. It also includes updating the system to make improvements. Users and technical specialists will go through a formal post-implementation audit that determines how well the new system has met its original objectives and whether any revisions or modifications are required. After the system has been fine-tuned it will need to be maintained while it is in production to correct errors, meet requirements, or improve processing efficiency. Over time, the system may require so much maintenance to remain efficient and meet user objectives that it will come to the end of its useful life span. Once the system’s life cycle comes to an end, a completely new system is called for and the cycle may begin again.

14.5 SYSTEMS ANALYSIS

Systems analysis is the analysis of the problem that the organization will try to solve with an information system. It consists of defining the problem, identifying its causes, specifying the solution, and identifying the information requirements that must be met by a system solution.

The key to building any large information system is a thorough understanding of the existing organization and system. Thus, the systems analyst creates a road map of the existing organization and systems, identifying the primary owners and users of data in the organization. These stakeholders have a direct interest in the information affected by the new system. In addition to these organizational aspects, the analyst also briefly describes the existing hardware and software that serve the organization.

From this organizational analysis, the systems analyst details the problems of existing systems. By examining documents, work papers, and procedures; observing system operations; and interviewing key users of the systems, the analyst can identify the problem are and objectives to be achieved by a solution. Often the solution requires buildings a new information system or improving an existing one.

Feasibility

In addition to suggesting a solution, systems analysis involves a feasibility study to determine whether that solution is feasible, or achievable, given the organization's resources and constraints. Three major areas of feasibility must be addressed:

Technical Feasibility: Whether the proposed solution can be implemented with the available hardware, software, and technical resources.

Economic Feasibility: Whether the benefits of the proposed solution outweigh the costs. We explore this topic in greater detail in Section 11.4, Understanding the Business Value of Information Systems.

Organizational Feasibility: Whether the proposed solution is desirable within the existing managerial and organizational framework.

Normally the systems analysis process will identify several alternative solutions that can be pursued by the organization. The process will then assess the feasibility of each. Three basic solution alternatives exist for every systems problem:

1. To do nothing, leaving the existing situation unchanged
2. To do modify or enhance existing systems
3. To develop a new system

There may be several solution design options within the second and third solution alternatives. A written systems proposal report will describe the costs and benefits, advantages and disadvantages of each alternative. It is then up to management to determine which mix of costs, benefits, technical features, and organization impacts represents the most desirable alternative.

Establishing Information Requirements

Perhaps the most difficult task of the systems analyst is to define the specific information requirements that must be met by the system solution selected. This is the area where many large system efforts go wrong and the one that poses the greater difficulty for the analyst. At the most basic level, the information requirements of a new system involve identifying who needs what information, where, when, and how. Requirements analysis carefully defines the objectives of the new or modified system and develops a detailed description of the functions that the new system must perform. Requirements must consider economic, technical, and time constraints, as well as the goals, procedures, and decision processes of the organization. Faulty requirements analysis is a leading cause of systems failure and high systems development costs. A system designed around the wrong set of requirements either will have to be discarded because of poor performance or will need to be heavily revised. Therefore, the importance of requirements analysis must not be underestimated.

Developing requirements specifications may involve considerable research and revision. A business function may be very complex or poorly defined. A manual system or routine set of inputs and outputs may not exist. Procedures may vary from individual to individual. Such situations will be more difficult to analyze, especially if the users are unsure of what they want or need (this problem is extremely common). To derive information systems requirements, analysts may be forced to work and re-work requirements statements in cooperation with users. Although this process is laborious, it is far superior to and less costly than redoing and undoing an entire system. There are also alternative approaches to eliciting requirements that help minimize these problems.

In many instances, business procedures are unclear or users disagree about how things are done and should be done. Systems analysis often makes an unintended contribution to the organization by clarifying procedures and building organizational consensus about how things should be done. In many instances, building a new system creates an opportunity to redefine how the organization conducts its daily business.

Some problems do not require an information system solution, but instead need an adjustment in management, additional training, or refinement of existing organizational procedures. If the problem is information-related, systems analysis may still be required to diagnose the problem and arrive at the proper solution.

14.6 SYSTEMS DESIGN

While systems analysis describes what a system should do to meet information requirements, systems design shows how the system will fulfill this objective. The design of an information system is the overall plan or model for that system. Like the blueprint of a building or house, it consists of all the specifications that give the system its form and structure. Information systems design is an exacting and creative task demanding imagination, sensitivity to detail, and expert skills.

Systems design has three objectives. First, the systems designer is responsible for considering alternative technology configurations for carrying out and developing the system as described by the analyst. This may involve analyses of the performance of different pieces of hardware and software, security capabilities of systems, network alternatives, and the portability or changeability of systems hardware.

Second, designers are responsible for the management and control of the technical realization of systems. Detailed programming specifications, coding of data, documentation, designers are responsible for the actual procurement of the hardware, consultants, and software needed by the system.

Third, the systems designer details the system specifications that will deliver the functions identified during systems analysis. These specifications should address all of the managerial, organizational, and technological components of the system solution.

14.6.1 Logical and Physical Design

The design for an information system can be broken down into logical and physical design specifications. Logical design lays out the components of the system and their relationship to each other, as they would appear to users. It showed what the system solution would do as opposed to how it is actually implemented physically. It describes inputs and outputs, processing functions to be performed, business procedures, data models, and controls. (Controls specify standards for acceptable performance and methods for measuring actual performance in relation to these standards. They are described in detail).

Physical design is the process of translating the abstract logical model into the specific technical design for the new system. It produces the actual specifications for hardware, software, physical databases, input/output media, manual procedures, and specific controls. Physical design provides the remaining specifications that transform the abstract logical design plan into a functions system of people and machines.

Design Alternative

Like houses or buildings, information systems may have many possible designs. They may be centralized or distributed, on-line or batch, partially manual, or heavily automated. Each design represents a unique blend of all of the technical and organizational factors that shape an information system. What makes one design superior to others is the ease and efficiency with which it fulfills user requirements within a specific set of technical, organizational, financial, and time constraints.

Before the design of an information system is finalized, analysts will evaluate various design alternatives. Based on the requirements definition and systems analysis, analysts construct high-level logical design models. They then examine the costs, benefits, strengths, and weaknesses of each alternative.

Illustrate design alternatives for a corporate cost system, which maintains data on the costs of various products produced by the corporation's operating units in various locations. The first alternative is a batch system that maximizes the efficiency and economy of computer processing but requires extensive manual preparation of data. The batch system requires the following steps:

1. Operating units prepare cost sheets with product cost data by plant. Sheets are mailed to corporate cost accounting at corporate headquarters.
2. Corporate cost accounting reviews cost sheets and prepares transactions forms, which are entered into the system.
3. The corporate product database is updated twice weekly via batch processing. The database maintains standard product cost data by plant and links local product numbers to corporate product numbers. The update also produces standard cost sheets.
4. Copies the standard cost sheets are mailed back to the operating units.

There is also a time lag between the preparation of operating unit cost sheets and the point when this information is reflected on the product database.

The second design alternative is an on-line system featuring more timely information and reduced manual effort, but at greater cost for computer processing, software, and security and recovery procedures required to maintain the integrity of the product database. The steps for the on-line system are as follows:

1. Operating units enter their own product cost data on-line via local CRT terminals with telecommunications links to the central corporate mainframe.

2. Through extensive on-line editing, the operating unit product data are edited. Errors are corrected and the data immediately update the corporate product database.

3. Up-to-date product cost information is available immediately after update. The system produces hard copy standard cost sheets or allows the operating units to perform on-line inquires about product cost information.

This alternative reduces manual activities and provides up-to-date-minute information both to corporate cost accounting and to the operating units.

The Role of End Users

Technical specialists cannot direct information systems design alone. It demands a very high level of participation and control by end users. User information requirements drive the entire systems-building effort. Users must have sufficient control over the design process to ensure that the system reflects their business priorities and information needs, not the biases of the technical staff.

Working on design increases users' understanding and acceptance of the systems, reducing problems caused by power transfers, inter-group conflict, and unfamiliarity with new system functions and procedures. Insufficient user involvement in the design effort is a major cause of system failure.

Some MIS researchers have suggested that design should be "user led." However, other researchers point out that systems development is not an entirely rational process. Users leading design activities have used their position to further private interests and gain power rather than to enhance organizational objectives. Users controlling design can sabotage or seriously impede the systems-building effort.

The nature and level of user participation in design vary from system to system. There is less need for user involvement in systems with simple or straightforward requirements than in those with requirements that are elaborate, complex, or vaguely defined. Transaction processing or operational control systems have traditionally required less user involvement than strategic planning, information reporting, and decision-support systems. Less structured systems need more user participation to define requirements and may necessitate many versions of design before specification can be finalized.

Different levels of user involvement in design are reflected in different systems development methods. How user involvement varies with each development approach.

14.7 IMPLEMENTATION AND MAINTENANCE

The remaining steps in the systems development process translate the solution specifications established during systems analysis and design into a fully operational information system. These concluding steps consist of programming, testing, conversion, and production and maintenance.

14.7.1 Programming

The process of translating design specifications into software for the computer constitutes a smaller portion of the systems development cycle than design and perhaps the testing activities. But it here, in providing the actual instructions for the machine, that the heart of the system takes shape. During the programming stage, system specifications that were prepared during the design stage are translated into program code. On the basis of detailed design documents for files, transaction and report layouts, and other design details, specifications for each program in the system are prepared.

Some systems development projects assign programming tasks to specialists whose work consists exclusively of coding programs. Other projects prefer programmer/ analysts who both design and program functions. Since large systems entail many programs with thousands – even hundreds of thousands – of lines of code, programming teams are frequently used. Moreover, even if an entire system can be programmed by a single individual, the quality of the software will be higher if it is subject to group review.

Exhaustive and thorough testing must be conducted to ascertain whether the system produces the right results. Testing answers the question, “Will the system produce the desired results under known conditions?”

The amount of time needed to answer this question has been traditionally underrated in systems project planning. As much as 50 per cent of the entire software development budget can be extended in testing. Testing is also time-consuming: Test data must be carefully prepared, results reviewed, and corrections made in the system. In some instances, parts of the system may have to be redesigned. Yet the risks of glossing over this step are enormous.

Testing information system can be broken down into three types of activities: Unit testing, or program testing, consists of testing each program separately in the system. While it is widely believed that the purpose of such testing each program separately in antee that programs is error free, this goal is realistically impossible. Testing should be viewed instead as a means of locating errors in programs, focusing on finding all the ways to make a program fail. Once pinpointed, problems can be corrected.

System testing the functioning of the information system as a whole. It tries to determine if discrete modules will function together as planned and whether discrepancies exist between the ways the system actually works and the way it was conceived. Among the areas examined are performance times, capacity for the storage and handling peak loads, recovery and restart capabilities, and manual procedures.

Acceptance testing provides the final certification that the system is ready to be used in a production setting. Systems tests are evaluated by users and reviewed by management. When all parties are satisfied that the new system meets their standards, the system is formally accepted for installation.

It is essential that all aspects of testing be carefully thought out and that they be as comprehensive as possible. To ensure this, the development team works with users to devise a systematic test plan. The test plan includes all of the preparations for the series of tests previously described.

The general condition being tested here is a record change. The documentation consists of a series of test-plan screens maintained on a database (perhaps a microcomputer database) that is ideally suited to this kind of application.

Users play a critical role in the testing process. They understand the full range of data and processing conditions that might occur within their system. Moreover, programmers tend to be aware only of the conditions treated in their programs; the test data they devise are usually too limited. Therefore, input from other team members and users will help ensure that the range of conditions included in the test data is complete. Users can identify frequent and less common transactions, unusual conditions to anticipate, and most of the common types of errors that might occur when the system is in use. User input is also decisive in verifying the manual procedures for the system.

14.7.3 Conversion

Conversion is the process of changing from the old system to the new system. It answers the question, “Will the new system work under real conditions?” Four main conversion strategies can be employed: the parallel strategy, the direct cutover strategy, the pilot study strategy, and the phased approach strategy.

In a parallel strategy, both the old system and its potential replacement are run together for a time until everyone is assured that the new one functions correctly. This is the safest conversion approach because, in the even to errors or processing disruptions, the old system can still be used as a backup. However, this approach is very expensive, and additional staff or resources may be required to run the extra system.

The direct cutover strategy replaced the old system entirely with the new system on an appointed day. At first glance, this strategy seems less costly than parallel conversion strategy. However, it is a very risky approach that can potentially be more costly than parallel activities if serious problems with the new system are found. There is no other system to fall back on. Dislocations, disruptions, and the cost of corrections may be enormous.

The pilot study strategy introduces the new system only to a limited area of the organization, such as a single department or operating unit. When this pilot version is complete and working smoothly, it is installed throughout the rest of the organization, either simultaneously or in stages.

The phased approach strategy introduces the new system in stages, either by functions or by organizational units. If, for example, the system is introduced by functions, a new payroll system might begin with hourly workers who are paid weekly, followed six months later by adding salaried employees who are paid monthly to the system. If the system is introduced by organizational units, corporate headquarters might be converted first, followed by outlaying operating units four months later.

A formal conversion plan provides a schedule of all the activities required to install the new system. The most time-consuming activity is usually the conversion of data. Data from the old system must be transferred to the new system, either manually or through special conversion software programs. The converted data then must be carefully verified for accuracy and completeness.

Moving from an old system to a new one requires that end users be trained to use the new system. Detailed documentation showing how the system works from both a technical and end-user standpoint is finalized during conversion time for use in training and everyday operations. Lack of proper training and documentation contributes to system failure, so this portion of the systems development process is very important.

14.7.4 Production and Maintenance

After the new system is installed and conversion is complete, the system is said to be in production. During this stage, both users and technical specialists to determine how well it has met its original objectives and to decide whether any revisions or modifications are in order will review the system. Changes in hardware, software, documentation, or improve processing efficiency are termed maintenance.

Studies of maintenance have examined the amount of time required for various maintenance tasks. Approximately 20 per cent of the 20 per cent is concerned with changes in data, files, reports, hardware, or system software. But 60 percent of all maintenance work consists of making user enhancements, improving documentation, and recording system components for greater processing efficiency. The amount of work in the third category of maintenance problems could be reduced significantly through better system analysis and design practices.

14.8 SUMMARY

There is a fundamental dilemma faced by anyone developing a computer application. Most problems are so large they have to be split into smaller pieces. The difficulty lies in combining the pieces back into a complete solution. Often each piece is assigned to a different team, and sometimes it takes months to complete each section. Without a solid plan and control, the entire system might collapse. Thousands of system development projects have failed or been cancelled because of these complications.

Partly because of the problems that have been encountered in the past, and partly because of technological improvements, several techniques are available to develop computer systems. The most formal approach is known as the systems development life cycle. Most organizations have created their own versions of SDLC. Any major company that uses SDLC also has a manual that is several inches thick (or comparable online documentation) that lays out the rules that MIS designers have to follow. Although these details vary from firm to firm, all of the methods have a common foundation. The goal is to build a system by analyzing the business processes and breaking the problem into smaller, more manageable pieces.

14.9 UNIT END EXERCISES

1. What are the different phases of traditional system life cycle?
2. What are the three phases of traditional system life cycle where users are highly involved?
3. What are the various steps and deliverables in the development phase?
4. In which phase of system life cycle the following are performed? Defining the problem, identifying its causes, specifying the solution, and identifying the information requirements.
5. What are the three major areas of feasibility, which are addressed in system analysis?
6. Which of the design lays out the components of the system and their relationship to each other, as they would appear to users?

14.10 REFERENCES AND SUGGESTED FURTHER READINGS

Alter Steven (1999), *Information Systems (A Management Perspective)*, Pearson Education.

Boyd Donald, W (2001), *Systems Analysis and Modeling (A Macro-to-Micro Approach with Multidisciplinary Applications)* Academic Press and Harcourt India.

Ghezzi Carlo, Jazayeri Mehdi and Mandrioli Dino (1991), *Fundamentals of Software Engineering*, Prentice-Hall of India, New Delhi.

Pressman, Roger S. (2001), *Software Engineering (A Practitioner's Approach)*, McGraw-Hill, New Delhi.